安全服务链中虚拟网络功能分配与调度算法研究 *

黄　睿，张红旗

(解放军信息工程大学，郑州 450001)

摘　要：安全服务链中的虚拟网络功能（virtual network function，VNF）将传统网络安全功能与硬件设备解耦，使得服务功能的部署更具动态性和可扩展性。然而，VNF 向节点的合理分配以及节点上 VNF 的高效调度问题仍亟待解决。为此，基于软件定义网络（software defined network，SDN）和网络功能虚拟化（network function virtualization，NFV）环境，提出基于优化算法的解决方案。首先，对资源分配与调度问题进行举例并形式化定义问题的优化目标；其次，提出基于贪心算法的资源分配方案和基于混合蜂群算法的资源调度方案，统一协调解决 VNF 的资源分配与调度问题。最后，设计仿真实验，验证所提算法的时间复杂性和在总资源成本和总服务收益方面的提升；同时，对比混合蜂群算法和传统蜂群算法，结果显示前者具有更快的收敛速度。

关键词：安全服务链；虚拟网络功能；软件定义网络；网络功能虚拟化；贪心算法；混合蜂群算法

中图分类号：TP393　　doi: 10.3969/j.issn.1001-3695.2017.09.0949

Research on algorithm of VNF allocation and scheduling problems in security service chain

Huang Rui, Zhang Hongqi

(*People's Liberation Army Information Engineering University*, *Zhengzhou* 450001, *China*)

Abstract: The virtual network function in the security service chain decouples the traditional network security functions from the hardware devices, making the deployment of service functions more dynamic and extensible. However, the rational allocation of the VNF to the node and the efficient scheduling of the VNF on the node still need to be solved urgently. To this end, this paper presented a solution using optimization algorithm based on the software defined network and network function virtualization environment. First, this paper made an example of resource allocation and scheduling problem and formalizes the objective. Then, this paper proposed a resource allocation scheme based on greedy algorithm and a resource scheduling scheme based on hybrid bee colony algorithm to solve the problem coordinately. Finally, the simulation experiment is designed to verify the time complexity and the improvement of total resource cost and total service income of the proposed algorithm. Meanwhile, it compared the hybrid bee colony algorithm with the traditional bee colony algorithm, indicating that the former has better convergence rate.

Key Words: security service chain; virtual network function; software defined network; network function virtualization; greedy algorithm; hybrid bee colony algorithm

0　引言

当前网络安全服务依赖于网络运营商为其提供专用的硬件设备，加剧了网络安全功能与硬件设备的紧耦合关系，造成网络安全服务模式静态僵化，难以满足未来多样化的网络业务需求[1,2]。表现为：一方面现有安全功能（防火墙、IDS(intrusion detection system)、IPS(intrusion protection system)）的硬件化，导致安全功能封闭固化、灵活性差；另一方面安全功能的静态部署难以满足业务需求的动态变化，造成网络资源的浪费。因此，软件定义网络（software defined network, SDN）[3,4]下的安全服务链动态组合机制应运而生[5,6]。作为安全服务链构建部署过程中的重要组成部分，服务链中虚拟网络功能（virtual network function, VNF）[7]的分配与调度算法研究迫在眉睫。

当前，SDN 的迅猛发展催生了网络功能虚拟化（network function virtualization, NFV）[8,9]技术的蓬勃兴起。SDN 将控制功能从传统的分布式网络设备中解耦出来，并使其直接可编程，实现了网络的集中管控；NFV 通过虚拟化方式将以专有硬件形式部署的网络功能转变为在通用服务器上运行的 VNF，为上层网络功能创新提供了条件。两者结合为探索网络安全服务新模式提供了有力的支撑。

基金项目：国家“863”计划资助项目（2012AA012704）；郑州市科技领军人才项目（131PLJRC644）

作者简介：黄睿（1993-），女（回族），新疆乌鲁木齐人，硕士研究生，主要研究方向为网络安全（xjhr1009@163.com）；张红旗（1962-），男，教授，博导，主要研究方向为网络安全、等级保护、信息安全管理.

针对 VNF 资源分配与调度问题，现有相关研究较少且不全面。Riera 等人在文献[10,11]中首次对 VNF 资源调度问题进行了形式化的定义，将最小化完成时间作为某可行方案的优化目标，但并未给出具体的解决方法。Mijumbi 等人在文献[12]中基于禁忌搜索算法解决了 VNF 中的资源分配/映射问题，但并未针对资源调度问题给出合理化的解决方案。因此，本文通过借鉴这些新的研究思路，完善不足，创新性地提出了基于贪心算法的资源分配方案和基于混合蜂群算法的资源调度方案，将资源调度过程抽象建模为柔性作业车间调度问题（flexible Job-Shop scheduling problem, FJSP），并采用混合蜂群算法（hybrid bee colony, HBC）求解此问题，确保了资源调度过程的高效性和资源分配与调度过程的协调性。

1　问题描述

如图 1 所示，安全服务链中 VNF 物理资源配置问题可以分为两类：(1）虚拟网络嵌入（virtual network embedding, VNE）。VNE 是指虚拟设备嵌入到物理设备，主要涉及不同网络状态下物理节点的选择、可用物理资源的计算以及两类节点间的关联映射问题[13,14]。相关研究已经较为成熟，因此不在本文的探讨范围之内。（2）虚拟网络功能分配与调度（virtual network function allocation and scheduling, VNF-AS）。VNF-AS 是指根据不同安全服务需求对虚拟节点上的多个 VNF 进行合理分配与调度，优化执行顺序，降低资源损耗，力求产生最大能效。相关研究较少且不完善，因此成为本文关注的重点。

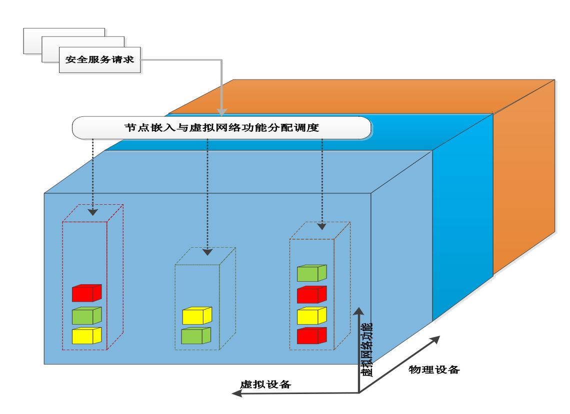


图 1　虚拟网络功能嵌入与调度问题

对于 VNF-AS 问题，存在许多资源共享的可能性。一种情况是，每一个 VNF 嵌入一个特定的虚拟设备，该虚拟设备的所有资源由其独享。例如，安全服务链 $SSC1: Firewall->IDS->Proxy->IPS->NAT$ 是由 5 个不同种类的 VNF 组成，这就意味着对于不同用户的安全服务请求，需要 5 个特定的虚拟设备为其提供服务。这不仅会对物理资源产生巨大的损耗，同时，也不宜于满足大规模网络中用户复杂的安全服务请求。本文采用基于容器技术实现的网络功能节点。如图 2 所示，高性能服务器（high volume server, HVS）作为提供安全服务的物理节点，HVS 中装配有容器集群，由编排器进行统一编排和管理。每个容器中，可开启多个 VNF，使得安全服务链中 VNF 的合理分配与优化调度成为可能。

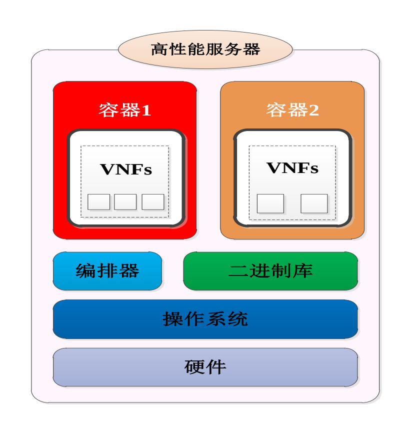


图 2　基于容器技术实现的网络功能节点

1.1　资源分配与调度举例

图 3～5 分别展示了两条不同的安全服务链 S_1、S_2 中 VNF 资源分配与调度情况。图 7 中 n_1-n_7 分别代表具有不同处理能力的 7 个网络节点，f_1-f_8 分别代表 8 个不同的 VNF。

a）T_1 时刻，安全服务请求 $S_1=\{f_8,f_2,f_3,f_6,f_5\}$ 到达，图 3 展示了一种可能的资源分配情况。节点 n_1 同时为虚拟网络功能 f_3 和 f_8 提供资源支持。同样地，假设在一段时间后，安全服务请求 $S_2=\{f_6,f_8,f_4\}$ 到达，其资源分配情况如图 4 所示。此时，两条安全服务链由虚拟节点同时进行处理。从图中可以观察到，S_1 和 S_2 的处理序列中都包含虚拟网络功能 f_6 和 f_8。对于 f_8 来说，分别交由节点 n_1 和 n_7 处理，无须等待，快速高效，而 f_6 则积压在节点 n_3 上，排队等待，造成不必要的处理时延。这就涉及 VNF 的资源优化分配问题，后文中将设计基于贪心算法的求解策略应对此问题。

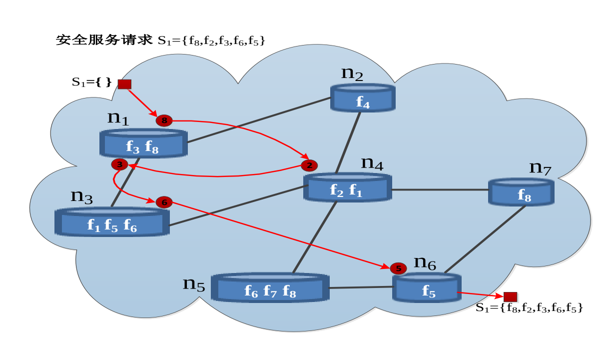


图 3　安全服务请求 S1 的资源分配

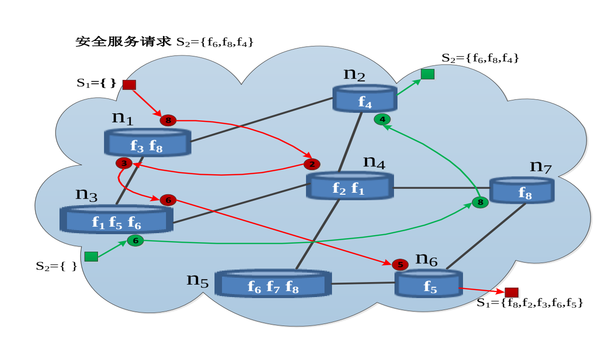


图 4　安全服务请求 S2 的资源分配

b）图 5 展示了安全服务链 S1 和 S2 的资源调度时间表。其中的每一个功能 f 必须按序处理，并且每一个节点在某一时段内只能处理一个特定的功能。定义符号 $P_{i,j}$ 代表 f_i 在虚拟节点 j 上的处理时延，其中 $1 \le i \le m$，$1 \le j \le n$，m 和 n 分别代表 f 的个数和虚拟节点的个数。本文采用假设的处理时间，表示每个节点上 f 的调度情况。在 T_1 时刻，安全服务请求 S_1 到达并立刻交由 n_1 节点进行处理。每一个后续功能需等到前序功能处理完成后才可以进行。在 T_8 时刻，S_1 处理完成。因此，S_1 处理时间为 $T_{S_1} = (T_8 - T_1)$，等价于链上各个 f 处理时间之合。为了阐明问题，假设安全服务请求 S_2 在 T_4 时刻到达。S_2 中的第一个 f 分配在节点 n_3，而此时节点 n_3 上已经分配了 S_1 的某个 f。因此，S_2 必须等到 T_6 时刻才可以开始工作。等待时间间隔 $(T_6 - T_4)$ 不仅增加了 S_2 的处理时间，还可能因 S_2 的其他 f 长时间空闲造成资源利用率的下降。这是由上一步资源分配导致的直接后果。假设将 S_2 中的 f_6 映射到节点 n_5，则可以有效避免资源调度过程中额外的时间开销。因此，统一协调资源分配与调度过程才能为安全服务提供保证。VNF 资源调度有三种可能的情况：(a) 在处理完一个功能后，立即处理下一个功能（图 5 中在 T_2 时刻，S_1 上的 f_2 紧接着 f_8 进行处理）；(b) 一个功能需等待上一个功能处理完之后才可进行处理（图 5 中 S_2 在 T_4 时刻到达，但需等待到 T_6 时刻，n_3 才能解除占用并为其提供服务）；(c) 一个功能需持续等待到一个链中所有功能处理完之后才能进行处理（图 5 中的 n_2 节点，需等待 $(T_9 - T_4)$ 的时间间隔才能完成处理）。后文中将资源调度问题抽象建模为 FJSP 并基于混合蜂群算法求解此问题。

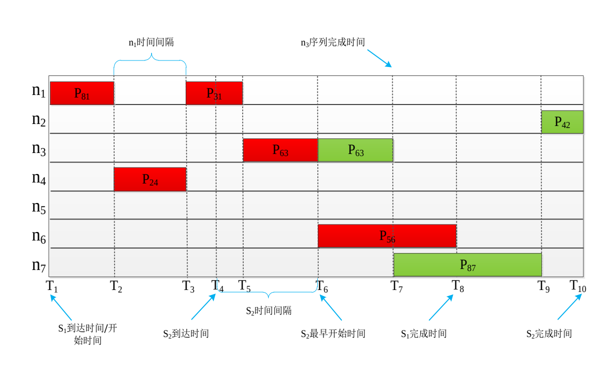


图 5　S1 和 S2 的资源调度时间表

1.2　资源分配与调度优化目标

在资源分配与调度问题中，有很多不同的优化目标。本文选取处理时间，资源成本和服务收益作为代表性指标进行分析。

1）处理时间　处理时间是指某一安全服务请求的最后一个功能处理完成到服务到达的时间间隔（式（1），t_e 代表完成时间，t_a 代表到达时间）。处理时间是两项参数的衡量指标：a）资源利用率（过高的处理时间说明服务长时间占据网络，造成较高的网络负载和资源消耗，资源利用率低下）；b）服务质量（处理时间越短，意味着服务的平均处理速度越快，服务质量越高）。

$$T = t_e - t_a \tag{1}$$

2)资源成本　资源成本是指对于某一特定的安全服务请求，物理网络资源的空间存储和时间处理开销。式（2）中，$\sum_{i=1}^{m} \delta_i$ 是指某一服务链 S 中所有功能 f 所耗费的空间存储资源之合，$(t_e - t_a)$ 代表时间处理开销，不仅包括节点的实际处理时间，也包括节点的等待时延，常量 α 和 β 分别代表两者的权重。

$$C = \alpha \sum_{i=1}^{m} \delta_i + \beta \left(t_e - t_a\right) \tag{2}$$

3）服务收益　服务收益是指物理网络资源为某一特定的安全服务请求提供服务的综合收益。式（3）中，布尔变量 $\gamma_{i,j}$ 值为 1 代表功能 f_i 分配在了节点 j 上，为 0 反之。变量 $\rho_{i,j}$ 则代表某一特定功能分配带来的服务收益。

$$R = \sum_{i=1}^{m} \sum_{j=1}^{n} \gamma_{i,j} \times \rho_{i,j} \tag{3}$$

考虑到三项指标具有不同的函数优化方向，定义效用函数 U 作为其综合性能反映，取值为所有性能参数指标经效用函数转换后的函数值。第 k 个方案的效用值计算如式（4）所示。

$$U_k = \omega_k \left[Y^+(r_+^k) + Y^-(r_-^k) + Y^s(r_s^k) \right],\ k = 1,2,\ldots,\kappa \tag{4}$$

其中：$\omega_k \ge 0$ 是缩放系数，用于调整各性能参数的影响因子；Y 为转换函数；r 是转换函数 Y 的自变量。第一个参数定义关于 r 的单增函数 Y^+，第二个参数定义关于 r 的单减函数 Y^-，对于时间性能参数，单独定义关于 r 的减函数 Y^s。此处，对性能参数进行归一化处理，得到取值在 $[0,1]$ 之间的归一化参数 $\tilde{r}^k$（式（5））。

$$\tilde{r}^k = \frac{r^k}{\max(r^k)},\ \tilde{r}^k \in [0,1],\ k = 1,2,\ldots,\kappa \tag{5}$$

参考效用函数理论，给出效用转换函数如式（6）所示。

$$T = \begin{cases} T^+ = e^{-(1-r^k)/r^k}, 0 < r^k < 1, k = 1,2,\ldots,\kappa \\ T^- = \begin{cases} e^{-r^k/(1-r^k)}, 0 < r^k < 1 \\ 0, r^k = 1 \end{cases}, k = 1,2,\ldots,\kappa \\ T^s = \cos(r^k \pi / 2), 0 < r^k \le 1, k = 1,2,\ldots,\kappa \end{cases} \tag{6}$$

2　基于贪心算法的资源分配方案

2.1　算法动机分析

本文提出一种针对 VNF 资源分配的贪心求解算法。根据不同安全服务请求，将算法中贪婪因子变量 H 划分为三种类型：a）最短处理时间（greedy least processing time, GLPT）；b）最快可用时间（greedy fast availability time, GFAT）；c）最大存储空间（greedy maximum memory space, GMMS）。其中，GLPT 是指将 VNF 分配至可提供最短处理时间的节点，该分配策略可满足基于处理时间计费的安全服务类型；GFAT 是指将 VNF 分配至可最快提供服务，等待队列最短的节点，该分配策略可满足迫切的用户安全服务需求；GMMS 是指将 VNF 分配至拥有

最大可用存储空间的节点，该分配策略可以平衡网络的实际负载，提升网络综合性能。

2.2　算法流程及算法描述

算法 1 给出了该算法的具体过程。可以看出该算法分为以下几步：

a）备份物理网络状态及函数初始化。为降低资源分配过程中因差错造成的损失，算法首先对物理网络状态进行备份并对安全服务链 S 、节点 N 和贪婪因子变量 H 进行初始化赋值。

b）扫描节点并分配资源。对于 N 中的任意一个节点 j，利用式（9）计算其处理结束时间，若节点空间存储和时间处理均满足约束条件（式（10）），则将该节点加入可用节点集合 N'，否则，扫描下一节点。待扫描完成后，判断集合 N'，如果恒为空，说明资源分配失败，重置备份的物理网络状态。

c）按贪婪因子排序。对不同的贪婪因子 H，进行节点排序，依次挑选序首节点进行资源分配，至此，算法结束。

算法 1　VNF 资源分配贪心求解算法

算法：贪心求解算法

输入：　S,N,H　。

输出：　$[f_i,n_j]$　。

(1)　*Start*

(2)　*Backup Substrate Network State*

(3)　*for VNF*　$f_i \in S$　*do*

(4)　　*Initialize: Capable Node Set*　$N' = \phi$　　*//初始化*

(5)　　*If(i==1) then*

　　$t_{i-1} = t_a$

(6)　　*end if*

(7)　　*for Node*　$j \in N$　*do*

(8)　　　$t_e = p_{i,j} + \max(\pi_j, t_{i-1})$　　*//计算结束时间*

(9)　　　*if* $((\gamma_{i,j} == 1) \wedge (\rho_{\max} \geq \rho_{i,j}) \wedge (t_e \leq t_l))$ *then*

　　　$N' = N' \uplus n_j$

(10)　　　*end if*

(11)　　*end for*

(12)　　*if*　$N' \equiv \phi$　*then*

(13)　　　*Resource Allocation Failed*

(14)　　　*Reset Substrate Network Status*

(15)　　　*return*

(16)　　*end if*

(17)　　*Sort*　N'　*according to*　H　*//按贪婪因子排序*

(18)　　*Select the top node*　n_j^*　*from*　N'

(19)　　*Allocation the function*　f_i　*onto*　n_j^*

(20)　　*Set*　$t_i = \max(\pi_j, t_{i-1})$

(21)　　*Update*　π_j　*and*　t_{i-1}

(22)　*end for*

(23)　*Resource Allocation Completed*

　End

3　基于混合蜂群算法的资源调度方案

3.1　算法动机分析

FJSP 是经典作业车间调度问题的扩展，是复杂的组合优化问题。FJSP 可描述为：加工系统有 n 个工件在 m 台设备上加工，每个工件包含 n_i 个预先确定加工顺序的工序，每个工序可在多台设备上加工，且加工时间随着设备性能的差别而不同。调度目标是在设备能力约束和工序约束下，将工序分配到设备，并确定在每台设备上工序的加工顺序。经对比分析，本文中 VNF 资源调度问题可以抽象建模为 FJSP，其数学模型涉及参数定义如下：

$M = \{M_k \mid 1 \leq k \leq m\}$，表示节点集;

$J = \{j_i \mid 1 \leq i \leq n\}$，表示安全服务链集;

$O_i = \{O_{ij} \mid 1 \leq j \leq n_i\}$，表示安全服务链 J_i 的服务功能集;

$M_{ij} = \{M_k \mid X_{ijk} = 1\}$，表示安全服务链 J_i 的服务功能 O_{ij} 的可用节点集;

M_{ijk} 表示服务功能 O_{ij} 在节点 M_k 上的加工时间;

S_{ijk} 表示服务功能 O_{ij} 在节点 M_k 上的开始加工时间;

E_{ijk} 表示服务功能 O_{ij} 在节点 M_k 上的完工时间;

FM_k 表示所有安全服务链在节点 M_k 上的完工时间;

FM 表示所有安全服务链的最后完工时间;

$$X_{ijk} = \begin{cases} 1 & \text{服务功能}O_{ij}\text{由节点}M_k\text{处理} \\ 0 & \text{其他} \end{cases};$$

$$R_{ijegk} = \begin{cases} 1 & O_{ij}\text{与}O_{eg}\text{同由节点}M_k\text{处理，}O_{ij}\text{先处理} \\ 0 & \text{其他} \end{cases};$$

以最大化效用函数值 U_k (式(4))为目标:

$$\max \sum_{k=1}^{K} U_K \tag{7}$$

对以上问题作以下约束:

a)互斥性约束

$$\begin{cases} E_{ijk} - E_{egk} \geq M_{egk} \\ R_{ijegk} = 1, X_{ijk} = X_{egk} = 1 \end{cases} \tag{8}$$

表示某个节点同一时间只能处理一个服务功能。

b)连续性约束

$$\begin{cases} E_{ijk} - S_{ijk} = M_{ijk} \\ X_{ijk} = 1 \end{cases} \tag{9}$$

表示服务功能的处理过程不可中断。

c)次序性约束

$$\begin{cases} S_{ijk} - E_{i(j-1)h} \geq 0 \\ X_{ijk} = X_{i(j-1)h} = 1 \end{cases} \tag{10}$$

表示同一安全服务链的服务功能间有顺序约束，不同安全服务

链间不存在服务功能顺序约束。

3.2　算法架构

HBC 的蜂群行为模型来自于 KARABOGA 建立的蜂群觅食行为的最小化模型[15]，包括食物源、雇佣蜂、观察蜂和侦察蜂四个组件。模型中，食物源代表问题的不同解，食物源的收益率（花蜜的数量）代表解得质量；雇佣蜂负责在一定时间内持续访问某个食物源及其附近区域；观察蜂负责收集雇佣蜂带来的食物源信息，以一定概率接纳对方并互相评估，保留优秀个体；侦察蜂负责随机搜索新的食物源，三类蜂群通过协作实现寻优。

图 6 为混合蜂群算法的主体框架。主算法分为种群初始化算法、雇佣蜂算法、观察蜂算法和侦察蜂算法四个部分，包括种群规模 SN 、迭代次数 $mcycle$ 、调节参数 rf 和领域规模 $nscale$ 四个参数。种群规模 SN 包含雇佣蜂和观察蜂两类，各占种群数量的一半；迭代次数 $mcycle$ 控制算法结束条件；调节参数 rf 用于协调算法的局部和全局搜索能力，初始值较小，算法过程中，随迭代次数的增加逐渐变大，以保证算法收敛性；领域规模 $nscale$ 控制领域解得产生范围，初始值较大，是为提高蜂群的解空间遍历能力，随迭代次数增加逐渐变小，目的是缩小搜索范围，对较优解进行精细搜索。

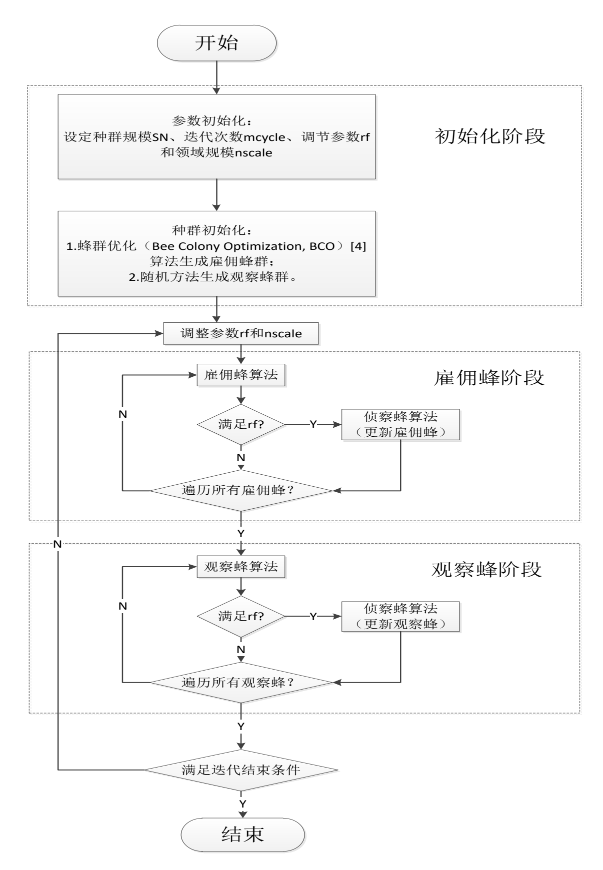


图 6　混合蜂群算法的主体框架

3.3　算法详细设计

3.3.1 种群初始化

应用于生产调度问题中的传统蜂群算法，多通过逐点遍历形成完整的调度方案，并进行全过程循环操作[16,17]。该方法计算量大，受种群规模影响严重，且逐点寻优易使算法早熟，陷入局部最优。因此，本文采用蜂群优化（bee colony optimization, BCO）[18]结合随机方法的方式，分别产生雇佣蜂和观察蜂初始种群，既提高了搜索效率，又增强了算法的全局搜索能力。

1）蜂群优化算法

BCO 算法主要包括前移和回顾两类操作。前移操作更新每个雇佣蜂编码，回顾操作根据概率丢弃部分雇佣蜂，并选择其他雇佣蜂替代。图 7 所示为 BCO 算法流程。其中 $step$ 的取值范围为 $1\sim SN/2$，当前雇佣蜂丢弃概率 P_{d_c} 取值如式（11）所示。

$$P_{d_c}=e^{-(fit_c-fit_{best})} \tag{11}$$

其中：fit_c 为当前雇佣蜂适应度；fit_{best} 为当前种群的最好适应度。最后生成雇佣蜂群 $ES\{X_i\}(i=1,2,\ldots,SN/2)$，并为 ES 初始化数组 $array1()$，记录 X_i 未更新的代数。

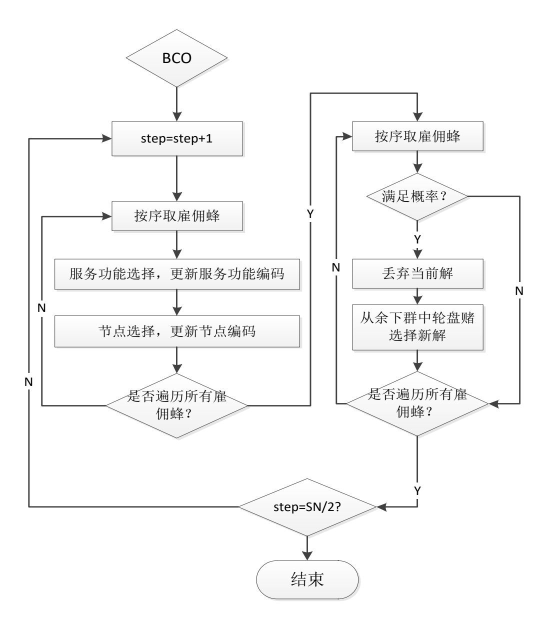


图 7　BCO 算法流程

2）随机方法

随机方法以随机数组和随机数组合的方式直接生成整段编码，步骤如下：

a)对每个观察蜂，将随机生成长度为服务功能数的数组作为服务功能段编码；

b)对于上述每一个服务功能，随机选择处理节点，组成节点段编码；

c)遍历观察蜂，建立种群 $OLS\{OL_i\}(i=1,2,\ldots,SN/2)$；

d)为 OLS 初始化数组 $array2()$，记录 OL_i 未更新的代数。

3.3.2 雇佣蜂觅食算法

雇佣蜂通过在领域内觅食产生新解。针对 FJSP 问题，本文采用改进领域解生成方法更新种群，建立雇佣蜂觅食算法 $Foraging()$,步骤如下：

a) $\forall X_i(i=1,2,\ldots,SN/2)$，$\mathrm{Neighbor}(X_i)$ 生成领域解 X_j；

b) $\forall X_j(j=1,2,\ldots,SN/2)$，比较 X_i 和 X_j 的适应度值，进行贪婪选择；

c)更新 $array1()$，若满足调节参数 rf，则执行侦察蜂算法 $Scout(X_i)$。

领域解更新算法 $Neighbor()$ 根据领域规模 $nscale$，随机选取 X_i 上两对服务功能码位进行交换，将互换后的服务功能重新选择分配节点，并重复上述过程 $nscale$ 次。

3.3.3 观察蜂算法

采用观察蜂算法，在两种蜂群中进行信息交换。对交互后的观察蜂群采用领域更新算法 $Neighbor()$，不断缩小领域范围，提高算法精度。建立观察蜂算法 $Observer()$，步骤如下：

a) $\forall OL_i(\mathrm{i}=1,2,...,\mathrm{SN}/2)$，根据适应度轮盘赌选择 X_i。

b)如果 X_i 较优，则以 X_i 替换 OL_i。

c)如果 OL_i 较优，则根据式（11）计算的概率选择是否保留 X_i，其中 fit_{best} 为 OL_i 适应度， fit_c 为 X_i 适应度。

d)更新 $array1()$ 和 $array2()$。

e)如果 $array1$ 满足 rf 条件，则对 X_i 进行侦察蜂操作 $Scout(X_i)$；如果 $array2$ 满足 rf 条件，则对 OL_i 进行侦察蜂操作 $Scout(OL_i)$。

f)对 OL_i 执行领域更新算法 $Neighbor(OL_i)$，生成领域解 $OL_i^*=Neighbor(OL_i)$。

g)评估领域解，判断是否更新 OL_i。

3.3.4 侦察蜂算法

侦察蜂算法 $Scout()$ 由调节参数 rf 触发，当雇佣蜂 X_i 或观察蜂 OL_i 对应的 $array$ 值等于 rf 时，产生随机解替代原解。随机解采用随机种群的初始化方法生成。

4　仿真实验与结果分析

4.1　实验环境

为验证算法效果，利用 Java 语言开发独立事件模拟器，通过 MATLAB 软件评估本文所提算法。每次资源分配和调度事件均服从泊松分布。本文定义 10 个不同的网络功能并将其编号为 1-10，每一个安全服务请求都由其中的一个或多个功能组成。设定每 5 个时间单元到达一项安全服务，且每项服务功能独占一个节点的资源直至处理完成。除非特殊说明，实验中的参数均在满足均匀分布的条件下从最大最小值之间随机抽取（表 1）。每个独立的实验过程涉及安全服务链到达、资源分配和调度以及处理结束后服务离开三个步骤。

表 1　实验参数范围

参数（单位）	最小值	最大值
节点数（个）	10	50
节点存储空间（兆）	10	20
单个节点处理功能数（个）	1	7
单个功能处理时间（毫秒）	100	500
单个功能存储空间需求（兆）	1	2
安全服务链长度（个）	3	10
安全服务处理时间（毫秒）	1000	5000

4.2　算法测试及结果分析

考虑到 VNF 资源分配与调度是一个连续的过程，资源分配的结果将直接影响资源调度的性能。因此，本文对贪心算法（GLPT、 GFAT 和 GMMS）和混合蜂群算法（HBC）进行组合测试。同时，为验证种群初始化方法对本文所提混合蜂群算法性能方面的提升，将其与传统蜂群算法对比进行验证。实验结果如图 8～12 所示。

1）平均等待时间和平均处理时间实验结果分析　图 8 和 9 分别展示了由独立事件模拟器产生的 1 000 条安全服务链的平均等待时间和平均处理时间。等待时间是指某一服务功能到达直到其开始进行处理的这一段时间间隔，平均等待时间是对某一安全服务的所有服务功能的等待时间加合求均值。完成时间是由所有服务功能处理时间和服务等待时间共同决定的。从图 8 中可以观察到，GLPT-HBC 算法相较于其他算法平均等待时间较长，这是因为 GLPT 算法倾向于将服务功能分配在能够最快处理的节点上，这将导致拥有最短处理时间的节点过载，易出现排队等候，从而造成较高的平均等待时间。GMMS-HBC 算法倾向于将服务功能分配在拥有最低负载的节点上，降低了服务执行和等待时间。GFAT-HBC 算法相较于其他算法具有最佳的时间性能表现，是因为 GFAT-HBC 算法的优化目标就是最小化处理完成时间，而等待时间恰是处理时间中的一部分。因此，图 9 中的实验结果与图 8 具有相似性和一致性。

2）总资源成本和总服务收益实验结果分析　图 10 和 11 分别展示了资源分配和调度的总资源成本和总服务收益。可以看出图中的数值是不断累积的，这是因为每经过一次成功的资源分配与调度，都会计算一次资源成本（式（2））和服务收益（式（3）），并累加在先前的数值之上。从图 10 中可以观察到，GLPT-HBC 算法资源成本最高，GFAT-HBC 算法资源成本最低，并且三种算法增长速率几乎相同。造成三者之间轻微差异的原因在于三种算法的平均等待时间不同。从图 11 中可以观察到，随着安全服务请求的增长，GFAT-HBC 算法的服务收益不断增加且高出另外两种算法约 40%，而 GMMS-HBC 和 GLPT-HBC 算法的增长速率几乎相同。从长远角度来看，GFAT-HBC 算法可为服务供应商提供更大的收益。

3）蜂群算法比较分析　图 12 对比了仅采用随机方法生成初始种群的传统蜂群算法和采用种群初始化方法生成种群的混合蜂群算法。算法均在种群规模 50、迭代次数 50、其他参数一致的情况下进行求解，且均求解出最优解 7。经对比分析，本文所提混合蜂群算法在提高初始解的质量的同时提高了算法的收敛速度。

5　结束语

针对安全服务链中 VNF 资源分配与调度问题，本文分别基于贪心算法和混合蜂群算法提出解决方案，设计了三种不同应用场景下的贪心算法以应对用户多样化的安全服务需求，仿真实验结果验证了三种算法不同的处理能力，创新性地提出了

基于种群初始化方法的混合蜂群算法解决 VNF 调度问题。实验结果表明，该算法相较于传统蜂群算法具有更快的收敛速度，适于满足大规模的网络需求。目前，安全服务链中虚拟网络功能的相关研究正不断兴起，下一步将继续完善本文研究，并深入开展服务链实验验证。

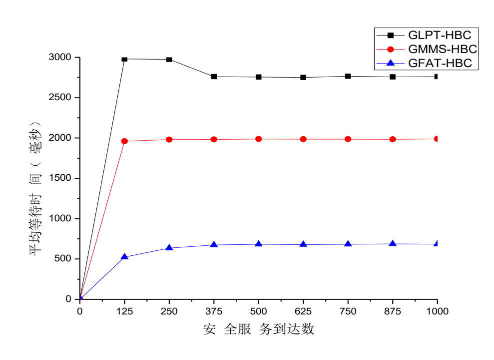


图 8　平均等待时间

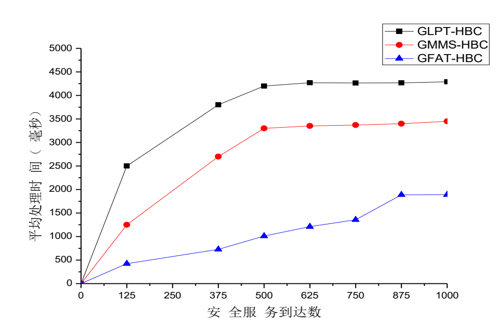


图 9　平均处理时间

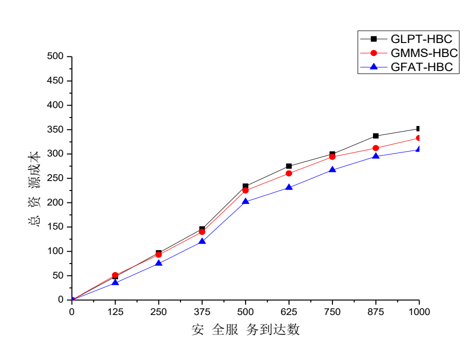


图 10　总资源成本

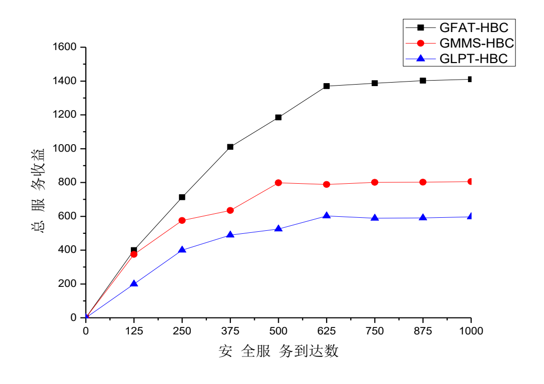


图 11　总服务收益

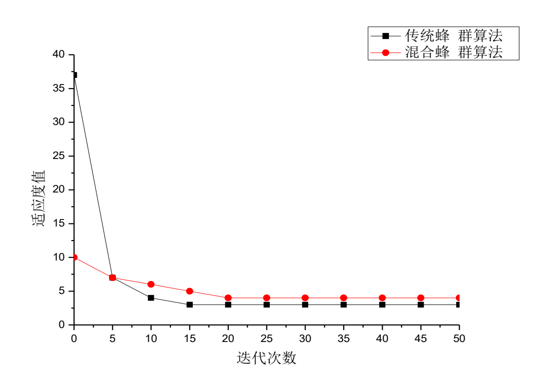


图 12　蜂群算法比较

参考文献：

[1] 黄韬，刘江，霍如，等．未来网络体系架构研究综述 [J]．通信学报，2014, 35 (8): 184-197.

[2] 兰巨龙，程东年，胡宇翔．可重构信息通信基础网络体系研究 [J]．通信学报，2014 (1): 128-139.

[3] Dave T. OpenFlow: enabling innovation in campus networks [J]. ACM Sigcomm Computer Communication Review, 2008, 38 (2): 69-74.

[4] 左青云，陈鸣，赵广松，等．基于 OpenFlow 的 SDN 技术研究 [J]．软件学报，2013 (5): 1078-1097.

[5] 熊钢，胡宇翔，段通，等．一种软件定义网络的安全服务链动态组合机制 [J]．电子与信息学报，2016, 38 (5): 1234-1241.

[6] Liu J, Li Y, Wang H, et al. Leveraging software-defined networking for security policy enforcement [J]. Information Sciences An International Journal, 2016, 327 (C): 288-299.

[7] Clayman S, Maini E, Galis A, et al. The dynamic placement of virtual network functions [C]// Proc of IEEE Network Operations and Management Symposium. 2014: 1-9.

[8] CHIOSI M, CLARKE D, WILLIS P, et al. Network functions virtualization-introductory white paper [C]// Proc of SDN and OpenFlow World Congress. 2012. : 1-5.

[9] Herrera J G, Botero J F. Resource allocation in NFV: a comprehensive survey [J]. IEEE Trans on Network & Service Management, 2017, 13 (3): 518-532.

[10] Riera J F, Hesselbach X, Escalona E, et al. On the complex scheduling formulation of virtual network functions over optical networks [C]// Proc of IEEE International Conference on Transparent Optical Networks. 2014: 1-5.

[11] Ferrer Riera, J, Escalona, E, Batalle, J, et al. Virtual network function scheduling: concept and challenges [C]// Proc of IEEE International Conference on Smart Communications in Network Technologies. 2014: 1-5.

[12] Mijumbi R, Serrat J, Gorricho J L, et al. Design and evaluation of algorithms for mapping and scheduling of virtual network functions [C]// Proc of IEEE Network Softwarization. 2015: 1-9.

[13] Fischer A, Botero J F, Beck M T, et al. Virtual network embedding: a survey [J]. IEEE Communications Surveys & Tutorials, 2013, 15 (4): 1888-1906.

[14] Rabbani M G, Pereira E R, Podlesny M, et al. On tackling virtual data center embedding problem [C]// Proc of IFIP//IEEE International Symposium on Integrated Network Management. 2013: 177-184.

[15] Karaboga D. An idea based on honey bee swarm for numerical optimization, Technical Report-TR06 [R]. 2005.

[16] Chong C S, Low M Y H, Sivakumar A I, et al. Using a Bee colony algorithm for neighborhood search in Job-Shop scheduling problems [C]// Proc of European Conference on Modelling & Simulation: Simulations in United Europe. 2008.

[17] Wong L P, Chi Y P, Low M Y H, et al. Bee colony optimization algorithm with big valley landscape exploitation for Job-Shop scheduling problems [C]// Proc of Conference on Winter Simulation. Winter Simulation Conference. 2008: 2050-2058.

[18] Teodorovic D, Lucic P, Markovic G, et al. Bee colony optimization: principles and applications [C]// Proc of IEEE Seminar on Neural Network Applications in Electrical Engineering. 2007: 151-156.